A Spectral Co-Clustering Approach for Dynamic Data

Derek Greene, Pádraig Cunningham
Clique Research Cluster, School of Computer Science & Informatics,
University College Dublin, Ireland
{derek.greene,padraig.cunningham}@ucd.ie

University College Dublin
Technical Report UCD-CSI-2011-08
August 2011

Abstract

A common task in many domains with a temporal aspect involves identifying and tracking clusters over time. Often dynamic data will have a feature-based representation. In some cases, a direct mapping will exist for both objects and features over time. But in many scenarios, smaller subsets of objects or features alone will persist across successive time periods. To address this issue, we propose a dynamic spectral co-clustering algorithm for simultaneously clustering objects and features over time, as represented by a set of related bipartite graphs. We evaluate the algorithm on several synthetic datasets, a benchmark text corpus, and social bookmarking data.

1 Introduction

In many domains, where the data has a temporal aspect, it will be useful to analyse the formation and evolution of patterns in the data over time. For instance, researchers may be interested in tracking evolving communities of social network users, such as clusters of frequently interacting authors in the blogosphere, or circles of users with shared interests on social media sites. In bibliometrics, this may include the analysis of the evolution of research communities within and across academic disciplines. In the case of online news sources, producing large volumes of articles on a daily basis, it will often be useful to chart the development of individual news stories over time.

For many of these problems it may be of interest to simultaneously identify clusters of both data objects and features. This task, often referred to as *co-clustering*, has been formulated as the problem of partitioning a bipartite graph, where the two types of nodes correspond to objects and features (Dhillon, 2001). This work has been almost entirely limited to static data exploration applications, where temporal information is unavailable or has been disregarded.

A popular recent approach to the problem of clustering dynamic data has been to use a step-based strategy, where the dynamic data is divided into discrete *time steps* of fixed duration. Sets of *step clusters* are identified on the individual time step datasets using a suitable clustering algorithm, and these step clusters are associated with one another over successive time steps (Tantipathananandh *et al.*, 2007). However, clusters may change considerably between time steps. This can be problematic, both for the purpose of matching clusters between time steps, and for supporting users to follow and understand how groups are changing over time. To address this problem, both current and historic information can be incorporated into the objective of the clustering process (Chakrabarti *et al.*, 2006). Benefits of this approach include increasing the smoothness of transitions between cluster-

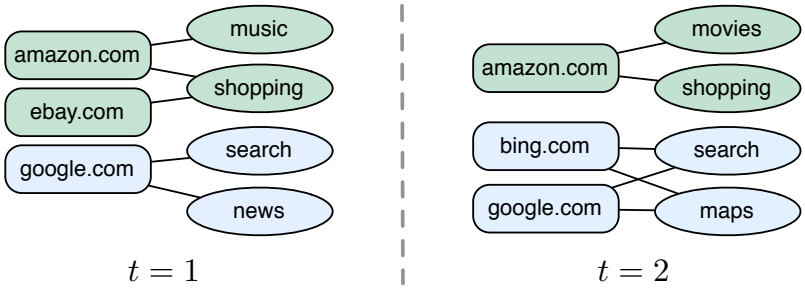


Figure 1: A dynamic co-clustering scenario where two clusters appear in successive time steps. Note that a subset of both objects (bookmarks) and features (tags) persists across time.

ings over time, and improving cluster quality by incorporating historic information to reduce the effects of noisy data.

A number of additional considerations arise when tracking dynamic data represented in feature spaces. Notably, a set of objects or features will not always persist in the data across steps. In general, three different scenarios are possible:

1. Data objects alone persist across time steps. For instance, in bibliographic networks, papers are only published at a single point in time, whereas authors will generally be present in the network over an extended period of time.
2. Features alone persist across time. In a news collection, articles will appear once, whereas terms may continue to appear as topics extend over time.
3. Both objects and features persist across time. For example, in the case of Web 2.0 tagging portals, both the individual tags and the objects being tagged (*e.g.* bookmarks, images) will appear in multiple time steps. A simple example with just two clusters is shown in Figure 1.

Here we consider the problem of tracking nodes in multiple related dynamic bipartite graphs. In Section 3 we describe the main contribution of this paper – the Dynamic Spectral Co-Clustering algorithm (DSCC), which simultaneously groups objects and features over time, in any of the above scenarios. This algorithm takes into account information from the current time step, together with historic information from the previous step. While our focus is primarily on the co-clustering of individual time step graphs, we also discuss the issue of tracking clusters across time steps.

In our evaluations in Section 4, we show that the DSCC algorithm is effective both in the case where features alone persist over time, and when objects **and** features persist. These evaluations are performed on a range of different datasets: synthetic datasets containing embedded cluster structures, a labelled benchmark news corpus, and a social bookmarking collection from the *Del.icio.us* web portal. On the labelled data we examine the ability of the dynamic co-clustering approach to correctly identify ground truth groupings, to deal with change in cluster structure over time, and to increase smoothness in the transitions between time step clusterings. In the case of the unlabelled collection, we explore the effectiveness of the algorithm in helping us to locate stable clusters representing meaningful trending topics, reflecting user interests and activity.

The remainder of the paper is structured as follows. In the next section we provide a summary of existing work in the areas of co-clustering and clustering of dynamic data. In Section 3 we outline the proposed dynamic co-clustering algorithm. An evaluation of the operation of this method on synthetic data, labelled benchmark text data, and real-world Web 2.0 tagging data is provided in Section 4. The paper concludes with suggestions for plans for future work.

2 Related Work

2.1 Co-Clustering

In certain problems it may be useful to perform *co-clustering*, where both objects and features are assigned to groups simultaneously. Such techniques are related to the *principle of the duality of*

clustering objects and features, where a clustering of objects induces a clustering of features, while a clustering of features also induces a clustering of objects (Dhillon, 2001). One approach to the co-clustering problem is to view it as the task of partitioning a weighted bipartite graph. Dhillon (2001) proposed a spectral approach to approximate the optimal normalised cut of a bipartite graph, which was applied for document clustering. This involved computing a truncated singular value decomposition (SVD) of a suitably normalised term-document matrix, constructing an embedding of both terms and documents, and applying k-means to this embedding to produce a simultaneous k-way partitioning of both documents and terms. Mirzal & Furukawa (2010) provided a further theoretical grounding for spectral co-clustering, demonstrating that simultaneous row and column clustering is equivalent to solving the separate row and column clustering problems.

A number of other co-clustering approaches have been proposed, including an information theoretic formulation involving alternating between updating row and column clusterings (Dhillon *et al.*, 2003), and a range of methods for producing soft co-clusterings via matrix factorization (Lee & Seung, 1999).

2.2 Semi-Supervised Clustering

For some real-world data exploration tasks a limited degree of supervision may be available. This may not necessarily correspond to the traditional notion of a subset of labelled training examples. For instance, the supervision could be derived from user feedback regarding the relations between pairs of objects in a small subset of a given dataset. This information is often represented as a set of pairwise constraints, where each constraint indicates that a pair of objects should either always be assigned to the same cluster or should never be assigned to the same cluster. This form of supervision can be used to guide a traditional clustering algorithm, either by providing a good set of initial clusters (Basu *et al.*, 2002), by using a "learnable" similarity function that adapts based on a small amount of label information (Bilenko, 2003), or by modifying the objective function of the algorithm to incorporate constraint information (Tseng, 2007). In the latter case, this can take the form of an additional reward or penalty term that quantifies the level of agreement between the current cluster memberships and the background information – a well-known example of this is the PCKMeans algorithm introduced by Basu *et al.* (2004).

2.3 Dynamic Clustering

The general problem of identifying clusters in dynamic data has been studied by a number of authors. Early work on the unsupervised analysis of temporal data focused on the problems of topic tracking and event detection in document collections (Yang *et al.*, 1998). More recently, Chakrabarti *et al.* (2006) proposed a general framework for "evolutionary clustering", where both current and historic information was incorporated into the objective of the clustering process. The authors used this to formulate dynamic variants of common agglomerative and partitional clustering algorithms. In the latter case, related clusters were tracked over time by matching similar centroids across time steps. Two evolutionary versions of spectral partitioning for classical (unipartite) graphs were proposed by Chi *et al.* (2007). The first version (PCQ) involved applying spectral clustering to produce a partition that also accurately clusters historic data. The second version (PCM) involved measuring historic quality based on the chi-square distance between current and previous partition memberships. Both algorithms were applied to synthetic data and weekly blog data.

The application of dynamic clustering methods has been particularly prevalent in the realm of social network analysis, where the goal is to identify communities of users in dynamic networks. Palla *et al.* (2007) proposed an extension of the popular CFinder algorithm to identify community-centric evolution events in dynamic graphs, based on an offline strategy. This extension involved applying community detection to composite graphs constructed from pairs of consecutive time step graphs. Another life-cycle model was proposed by Tantipathananandh *et al.* (2007), where the dynamic community finding approach was formulated as a graph colouring problem. The authors proposed a heuristic solution to this problem, by greedily matching pairs of node sets between time steps. The problem of clustering data over time has also been considered in the temporal analysis domain. Kalnis *et al.* (2005) described a density-based clustering approach where clusters persist over time, despite continuous changes in cluster memberships. This corresponds closely to the "assembly line" dynamic clustering scenario described by Tantipathananandh *et al.* (2007).

Little work has been done in adapting co-clustering methods to dynamic data. Koutsonikola *et al.* (2008) considered the problem of co-clustering pairs of related time series datasets (*e.g.* news and market data) based on successive snapshots, in order to reveal dependencies between the datasets. Giannakidou *et al.* (2010) described a "time-aware" user-tag clustering approach for application to dynamic social bookmarking data. The approach involves constructing a user-tag similarity matrix that includes both semantic information from Wordnet to deal with synonymy and temporal user-tag assignment information from successive time intervals. The authors used this approach to identify communities of users on Flickr tagging data.

3 Methods

3.1 Problem Definition

Before describing our proposed algorithm, we frame the dynamic co-clustering problem. We represent a dynamic feature-based dataset as a set of l bipartite graphs $\{G_1, \ldots, G_l\}$. Each *step graph* G_t consists of two sets of nodes, representing the n_t data objects, and m_t features present in the data at time t. Edges exist only between nodes of different types, corresponding to non-zero feature values. We can conveniently represent each step graph using a feature-object matrix $\mathbf{A}_t$ of size $m_t \times n_t$.

In the "step-based" formulation of the dynamic co-clustering problem, the overall goal is to identify a set of *dynamic clusters* of objects and features, which appear in the data across one or more time steps. We refer to *step clusters* that are identified on individual step graphs, which represent specific observations of dynamic clusters at a given point in time.

The formulation therefore has two key requirements: a suitable clustering algorithm to cluster individual time step graphs (ideally in a way that incorporates historic information), and an approach to track these clusters across time steps. While our primary focus here is on the former aspect, in Section 3.3 we also briefly discuss the latter aspect.

3.2 Dynamic Spectral Co-Clustering

We now describe the Dynamic Spectral Co-Clustering (DSCC) algorithm that considers both historic information from the previous time step, and the internal quality of the clustering in the current time step. The algorithm consists of three phases: (1) spectral embedding of the matrix representation of a bipartite graph, (2) a cluster initialisation phase, and (3) a cluster assignment phase.

3.2.1 Spectral Embedding

Following the formulation for normalised cut optimisation via spectral co-clustering described by Dhillon (2001), for the feature-object matrix $\mathbf{A}_t$ at time step t, we construct the degree-normalised matrix

$$\hat{\mathbf{A}}_t = {\mathbf{D_1}}^{-\frac{1}{2}} \mathbf{A}_t {\mathbf{D_2}}^{-\frac{1}{2}} \tag{1}$$

where $\mathbf{D_1}$ and $\mathbf{D_2}$ are diagonal column and row degree matrices defined as:

$$[D_1]_{ii} = \sum_{j=1}^{n} \mathbf{A}_t(i,j)\,, \qquad [D_2]_{jj} = \sum_{i=1}^{m} \mathbf{A}_t(i,j) \tag{2}$$

We then apply SVD to $\hat{\mathbf{A}}_t$, computing the leading left and right singular vectors corresponding to the largest singular values. Following the choice made by many authors in the spectral clustering literature (*e.g.* Ng *et al.* (2001)), we use k_t dimensions corresponding to the expected number of clusters. Although the issue of selecting the number of clusters is not discussed in this paper, one potential approach is to choose k_t based on the eigengap method (Ng *et al.*, 2001). The truncated SVD yields matrices $\mathbf{U_{k_t}}$ and $\mathbf{V_{k_t}}$. A unified embedding of size $(m_t + n_t) \times k_t$ is constructed by normalising and stacking the truncated factors as follows:

$$\mathbf{Z}_t = \begin{bmatrix} {\mathbf{D_1}}^{-1/2}\mathbf{U_{k_t}} \\ {\mathbf{D_2}}^{-1/2}\mathbf{V_{k_t}} \end{bmatrix} \tag{3}$$

Prior to clustering, the rows of $\mathbf{Z}_t$ are subsequently re-normalised to have unit length, as proposed for spectral partitioning in Ng *et al.* (2001). This process provides us with a k_t-dimensional embedding of all nodes of both types in G_t.

3.2.2 Cluster Initialisation

At time step $t = 1$, we have no historic information. Therefore to seed the clustering process, we use a variant of orthogonal initialisation as proposed by Ng *et al.* (2001) for spectral graph partitioning. This operates using a "farthest-first" strategy as follows. The first cluster centroid is chosen to be the mean vector of the rows in $\mathbf{Z}_t$. We then repeatedly select the next centroid to be the row in $\mathbf{Z}_t$ that is closest to being 90° from those that have been previously selected. This process continues until k_t centroids have been chosen.

For each time step $t > 1$, we initialise using clusters from the previous time step. A simple approach is to map the clusters generated on the embedding for time $t-1$ to $\mathbf{Z}_t$. However, as noted previously, not all features and objects will persist between time steps. To produce an initial clustering at time t, we identify the intersection of the sets of nodes present in the graphs G_{t-1} and G_t. The clusters containing these nodes are mapped to the embedding $\mathbf{Z}_t$, and we compute the resulting centroids and normalise these centroids to unit length. If less than k_t centroids are produced, the remaining centroids are chosen from the rows of $\mathbf{Z}_t$ using orthogonal selection as above. We can then predict memberships for each unassigned row z_i of $\mathbf{Z}_t$, using a simple nearest centroid classifier to maximise the dot product similarity:

$$\max_{C \in C_t} \; z_i^{\mathsf{T}} \mu_c \tag{4}$$

where μ_c is the normalised centroid of cluster C_c. This classification procedure yields a predicted clustering for all rows in $\mathbf{Z}_t$ (*i.e.* a co-clustering of all objects and features present in G_t), which we denote $\mathcal{P}_t$.

3.2.3 Cluster Assignment

To recover a clustering from $\mathbf{Z}_t$, we apply a constrained version of k-means clustering to the rows of the embedding, which takes into account both the internal quality of the current partition, and agreement with the predicted partition $\mathcal{P}_t$. We distinguish the latter from the membership preservation objective described by Chi *et al.* (2007) – here we use predicted memberships for missing objects and features missing from the previous step.

As a measure of current cluster quality, we use vector-centroid similarities as in Eqn. 4. Historical quality is calculated based on the quantity $\text{pred}(\mathcal{P}_t, \mathcal{C}_t)$, which denotes the degree to which the predicted cluster assignments in $\mathcal{P}_t$ agree with those in the current clustering $\mathcal{C}_t$. To quantify this agreement, we use a variant of the pairwise *prediction strength* measure proposed by Tibshirani *et al.* (2001) for stability analysis:

$$\text{pred}(\mathcal{P}_t, \mathcal{C}_t) = \sum_{C \in \mathcal{C}_t} \frac{1}{|C|\,(|C|-1)} \sum_{(z_i, z_j) \in C} co(z_i, z_j) \tag{5}$$

The value $co(z_i, z_j) = 1$ if rows z_i and z_j were predicted to be co-assigned in $\mathcal{P}_t$, or $co(z_i, z_j) = 0$ if they were predicted be assigned to different clusters.

To combine both sources of information, the clustering objective then becomes a weighted combination of two objectives:

$$J(\mathcal{C}_t) = (1-\alpha) \cdot \left(\sum_{c=1}^{k} \sum_{z_i \in C_c} z_i^{\mathsf{T}} \mu_c \right) + \alpha \cdot (\text{pred}(\mathcal{P}_t, \mathcal{C}_t)) \tag{6}$$

This type of aggregation approach has been widely used for combining sources of information, such as in dynamic clustering Chakrabarti *et al.* (2006) and semi-supervised learning Basu *et al.* (2004). The *balance* parameter $\alpha \in [0, 1]$ controls the trade-off between the influence of historical information and the information present in the current spectral embedding. A higher value of α allows information from the previous time step to have a greater influence, yielding a smoother transition between clusterings at successive time steps. Naturally at time $t = 1$, the right-hand term in Eqn. 6 will be zero.

Eqn. 6 can be viewed as the standard spherical k-means objective (Dhillon & Modha, 2001), augmented by a constraint reward term. We can find a local solution for this problem by using an approach analogous to the semi-supervised PCKMeans algorithm proposed by Basu *et al.* (2004)

1. Build spectral embedding
   - Construct the normalised feature-object matrix $\hat{\mathbf{A}}_t = \mathbf{D_1}^{-\frac{1}{2}} \mathbf{A}_t \mathbf{D_2}^{-\frac{1}{2}}$.
   - Compute the embedding $\mathbf{Z}_t$ from the truncated SVD of $\hat{\mathbf{A}}_t$ according to Eqn. 3.
   - normalise the rows of $\mathbf{Z}_t$ to unit length.
2. Initialisation and prediction
   - If $t = 1$, apply orthogonal initialisation to select a set of k_t representative centroids from the representations of the objects in the embedded space.
   - For $t > 1$, recompute the k_{t-1} centroids based on last clustering but including only the embedding of the relevant set of objects/features in the current space.
   - If less than k_t centroids have been produced, select remaining centroids to be orthogonal to existing centroids.
   - If not all rows of the embedding have been assigned, apply nearest centroid classification to compute the predicted clustering $\mathcal{P}_t$.
3. Compute clustering
   - Apply constrained k-means to rows in $\mathbf{Z}_t$, initialised by centroids from the prediction $\mathcal{P}_t$ to produce a co-clustering $\mathcal{C}_t$.
4. Repeat from #1 until all l time steps have been processed.

Figure 2: Overview of the Dynamic Spectral Co-Clustering (DSCC) algorithm.

for clustering with pairwise constraints. Specifically, we apply an iterative k-means-like assignment process, re-assigning each row vector z_i from $\mathbf{Z}_t$ to maximise:

$$\max_{C \in C_t} \; (1 - \alpha) \cdot z_i^{\mathsf{T}} \mu_c + \alpha \cdot \text{pred}(z_i, C) \tag{7}$$

where the quantity $pred(z_i, C)$ represents the degree to which the predicted assignment for the row z_i in P_t agrees with the assignment of z_i to cluster C. This is given by the proportion of rows in C that were co-assigned with z_i in P_t:

$$\text{pred}(z_i, C) = \frac{1}{|C|\,(|C| - 1)} \sum_{(z_i, z_j) \in C} co(z_i, z_j) \tag{8}$$

Once the algorithm has converged to a local solution, $\mathcal{C}_t$ provides us with a k-way partitioning of all nodes in the graph G_t (*i.e.* features and objects). An overview of the complete DSCC algorithm is shown in Figure 2.

3.3 Tracking Clusters Over Time

In the previous section we focused on the problem of co-clustering individual time step graphs in a dynamic context. A related aspect of the step-based approach to dynamic clustering involves identifying *dynamic clusters* composed from chains of clusters linked across time steps. We suggest that previous frameworks for tracking evolving dynamic communities (Greene *et al.*, 2010; Tantipathananandh *et al.*, 2007) can be readily adapted to the dynamic bipartite case.

In brief, we construct a set of dynamic cluster timelines, each consisting of a set of clusters identified at different time steps and ordered by time. At each time $t > 1$ in the dynamic co-clustering process, we match the most recent observations associated with the existing dynamic cluster timelines with the output of DSCC in the current time step. Matches are made based on the step cluster memberships for subsets of objects and/or features persisting between pairs of consecutive steps, using a set matching measure, such as the Jaccard index (Jaccard, 1912), and a user-defined matching threshold $\theta \in [0, 1]$. After processing all l time steps, this matching procedure will result in a set of dynamic clusters persisting across multiple steps, each consisting of a timeline of step clusters produced by DSCC.

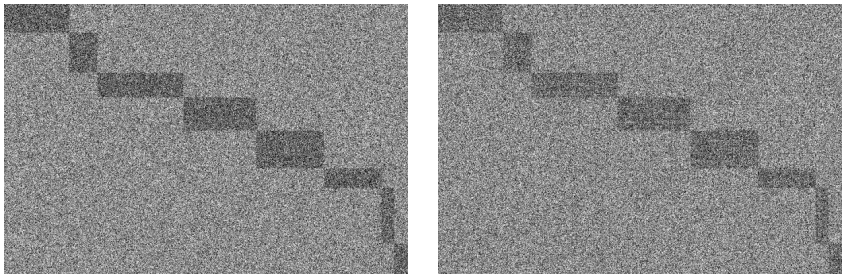
4 Evaluation

4.1 Synthetic Evaluation

To initially evaluate the behaviour of the DSCC algorithm proposed in Section 3.2, we conducted experiments on a number of dynamic synthetic datasets[1] containing embedded clusters. The goal of these evaluations was to examine the performance of DSCC when incorporating history data with different levels of volatility – from cases where clusters are stable over time, to cases where clusters change substantially between time steps.

4.1.1 Data Generation

Synthetic data was generated as follows. Each synthetic dataset consisted of l matrices, corresponding to l successive time steps. All features and objects persist across time steps. Each time step matrix $\mathbf{A}_t$ is rectangular, with n rows (features) and m columns (objects). These matrices contain k embedded rectangular structures, corresponding to clusters of both objects and features. For object i and feature j that are assigned to the same cluster, the corresponding entry $\mathbf{A}_t(i,j)$ will take a value 1 with probability p_{in}, and zero otherwise. For object i and feature j that are assigned to different clusters, the corresponding entry $\mathbf{A}_t(i,j)$ will take a value 1 with probability p_{out}, and zero otherwise. The latter entries correspond to the background noise in the data.



(a) $p_{swap} = 0.1,\ t = 1$ (b) $p_{swap} = 0.1,\ t = 2$

(c) $p_{swap} = 0.4,\ t = 1$ (d) $p_{swap} = 0.4,\ t = 2$

Figure 3: Matrices for the first two time steps from two synthetically-generated dynamic datasets with embedded clusters. The rows of the matrices are ordered according to cluster memberships at $t = 1$. The top pair of figures show a dataset that contains relatively little volatility ($p_{swap} = 0.1$). The dataset in the bottom figures contains a very high level of volatility ($p_{swap} = 0.4$).

At the first time step $t = 1$, objects and features were randomly assigned to the k clusters so that clusters were reasonably balanced in size, with $\pm 20\%$ random variation. After each time step, the cluster memberships of objects and features were swapped with a probability p_{swap}. This random permutation is intended to simulate the natural movement of nodes between clusters over time in a dynamic dataset. Once cluster memberships have been swapped, the next time step matrix $\mathbf{A}_{t+1}$ was constructed as described above.

For the evaluations described here, we constructed four datasets with $l = 10$ time steps, containing $n = 1,000$ objects and $m = 1,500$ features, assigned to $k = 8$ embedded clusters of objects and

[1] Datasets for this paper are available at `http://mlg.ucd.ie/dscc.html`

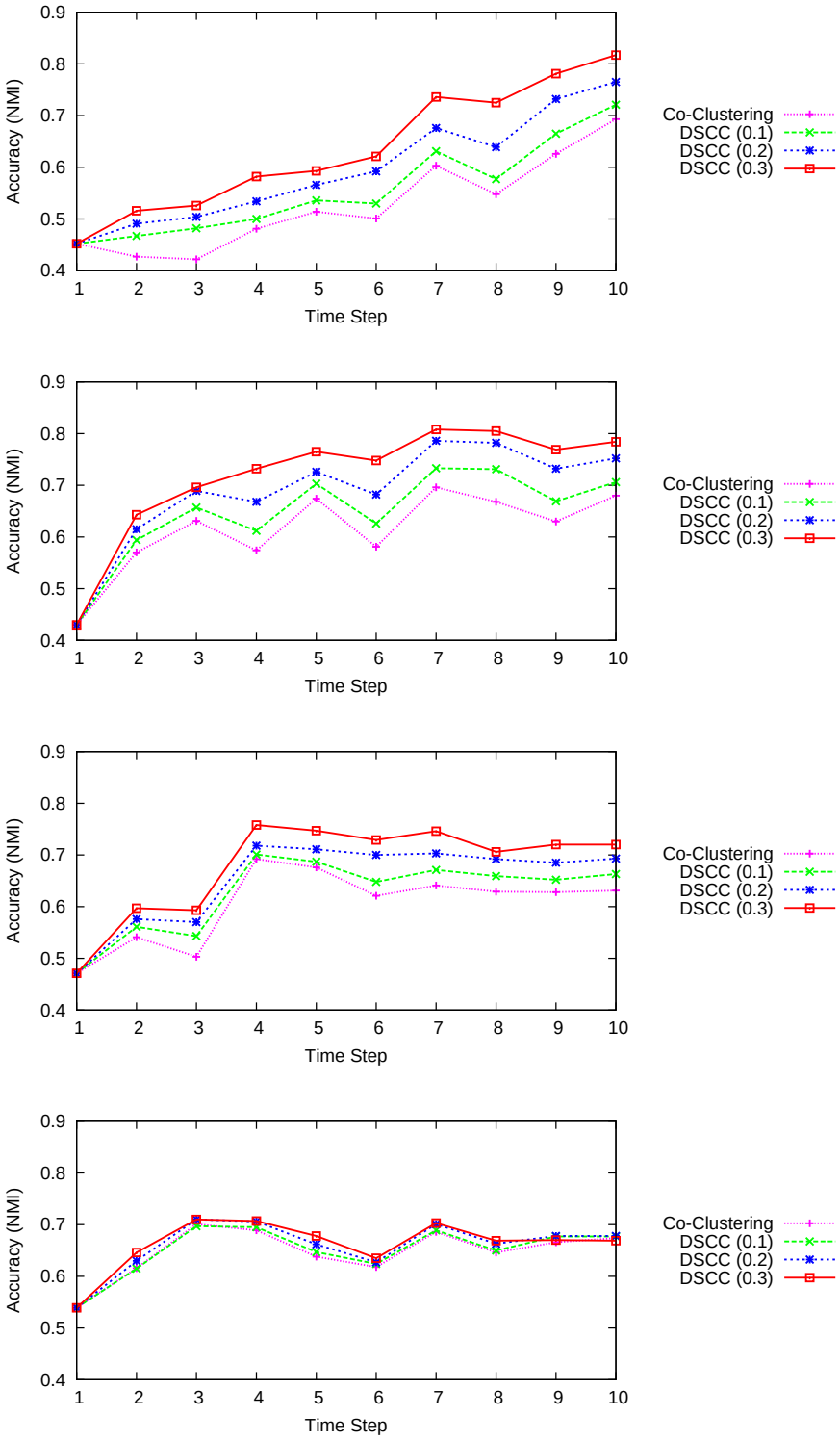


Figure 4: Comparison of accuracy (in terms of NMI) for **object clusterings** generated by standard spectral co-clustering and DSCC on four synthetic dynamic datasets generated with increasing levels of volatility $p_{swap} = \{0.1, 0.2, 0.3, 0.4\}$.

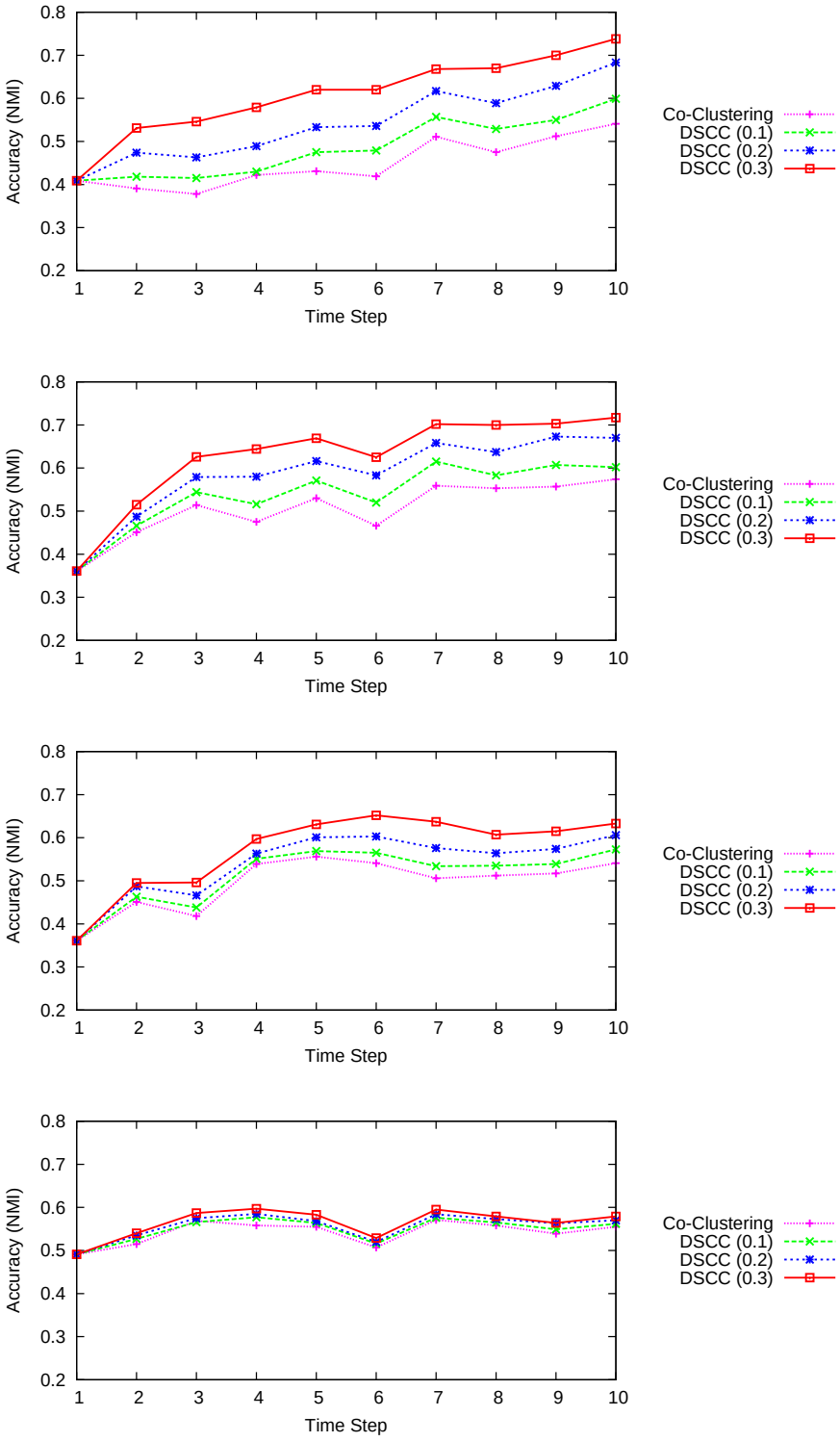


Figure 5: Comparison of accuracy (in terms of NMI) for **feature clusterings** generated by standard spectral co-clustering and DSCC on four synthetic dynamic datasets generated with increasing levels of volatility $p_{swap} = \{0.1, 0.2, 0.3, 0.4\}$.

features. To construct reasonably clear block cluster structures with some background noise, we set $p_{in} = 0.6$ and $p_{out} = 0.45$. Since the goal of our evaluation was to examine the degree to which the volatility impacted on the success of the DSCC algorithm, we focused on examining a range of values $p_{swap} \in [0.1, 0.4]$, where a higher value indicates a greater degree of cluster membership change between successive time steps. The relative difference in volatility between the first and last datasets is illustrated in Figure 3. We see that there is a substantially greater change in cluster memberships between time steps $t = 1$ and $t = 2$ for the dataset generated with $p_{swap} = 0.4$.

4.1.2 Discussion

We assessed the performance of DSCC at all time steps in each of the four dynamic synthetic datasets, using a range of values $\alpha \in [0.1, 0.3]$ for the balance parameter. As a baseline competitor, we used standard multi-partition spectral co-clustering as proposed by Dhillon (2001). To provide a fair comparison, we use orthogonal initialisation for both algorithms, and set the number of clusters to the number of embedded clusters $k = 8$. To quantify the performance of both algorithms, we measure the agreement between co-clustering and the embedded clusters (both object and feature clusters) in terms of their *normalised mutual information* (NMI) (Strehl & Ghosh, 2002).

Figure 4 illustrates the comparison of **object clustering** accuracy scores for DSCC and standard spectral co-clustering on the datasets, in increasing order of volatility. We observe that for the first dataset, where 10% of objects and features are swapped between clusters after each time step, a considerable increase in NMI is achieved by DSCC. This increase is more pronounced as α increases, so that historic data makes a greater contribution to the clustering objective defined in Eqn. 6. We see increases in the second and third datasets, where 20% and 30% of memberships are switched. For the highly volatile case of the fourth dataset, where 40% of memberships change between time steps, there is little improvement gained by incorporating temporal information – accuracy is comparable to that achieved by standard spectral co-clustering. Note that the average NMI between all pairs of embedded ground truth clusterings for this dataset is 0.04, and the agreement between the ground truth memberships at $t = 1$ and $t = 10$ is 0.01. This is little better the random, indicating that the cluster assignments have almost completely changed through the course of the dynamic process. Therefore it is unsurpising that DSCC does not lead to an improvement in accuracy here, given the level of volatility present in the data.

Figure 5 presents an analogous comparison of NMI scores for the **feature clusterings** generated on the ten successive time steps for each of the four synthetic datasets. We observed behaviour that is highly similar to that described above – DSCC leads to noticeable improvements in clustering accuracy, except in the case of extremely volatile data ($p_{swap} = 0.4$). Experiments on other synthetically-generated data, with a range of values for (p_{in}, p_{out}) for different levels of intra- and inter-cluster similarity, lead to very similar behaviour for both DSCC and standard co-clustering. In general, the results on synthetic data demonstrate that DSCC successfully allows us to use historic data to potentially improve clustering accuracy in cases where both objects and features persist across time.

4.2 Benchmark Evaluation

Next we evaluated the performance of DSCC on the dynamic bipartite document clustering problem. For this we required an annotated corpus with temporal information. We used a subset of the widely-used Reuters RCV1 corpus Lewis *et al.* (2004). The *RCV1-5topic* dataset consists of 10,116 news articles covering a seven month period. Each article is annotated with a single ground truth topical label: health, religion, science, sport, weather. These topics are present across the entire time period of the corpus. We considered a number of different time step durations to split the seven month period – one month, a fortnight, and one week – yielding 7, 14, and 28 step graphs respectively. Naturally for this type of data, a subset of features (terms) will persist across time, while objects (documents) appear in only one time step.

Our evaluations focused on the performance of DSCC on each time step graph in the *RCV1-5topic* dataset, using a range of values $\alpha \in [0.1, 0.3]$ for the balance parameter. Again we use multi-partition spectral co-clustering Dhillon (2001) with orthogonal initialisation as a baseline. We set the number of clusters k_t at each time step t to the number of ground truth topics in the data.

4.2.1 Temporal Smoothness

One of the primary motivations for dynamic co-clustering is to increase smoothness in the transitions between time step clusterings. To quantify the degree to which the proposed algorithm can enforce temporal smoothness, we measure the agreement between successive clusterings based on their pairwise NMI scores. Note that NMI values were calculated only over the terms common to each pair of consecutive time steps – documents are not considered as they do not persist.

Figure 6 shows a comparison of agreement values for the three different time window sizes. Dynamic co-clustering leads to a higher level of agreement than standard spectral co-clustering for all three time window sizes. The effect becomes significantly more pronounced as α increases, with a considerable rise apparent in Figure 6 at $\alpha = 0.3$. This is to be expected, as increasing the parameter leads to a higher weighting for the historic information in Eqn. 6. We also observed that, when we increase $\alpha \geq 0.4$, the resulting co-clusterings are often almost identical to the predicted co-clustering $\mathcal{P}_t$, with the constrained k-means process converging to a solution after 1–5 iterations.

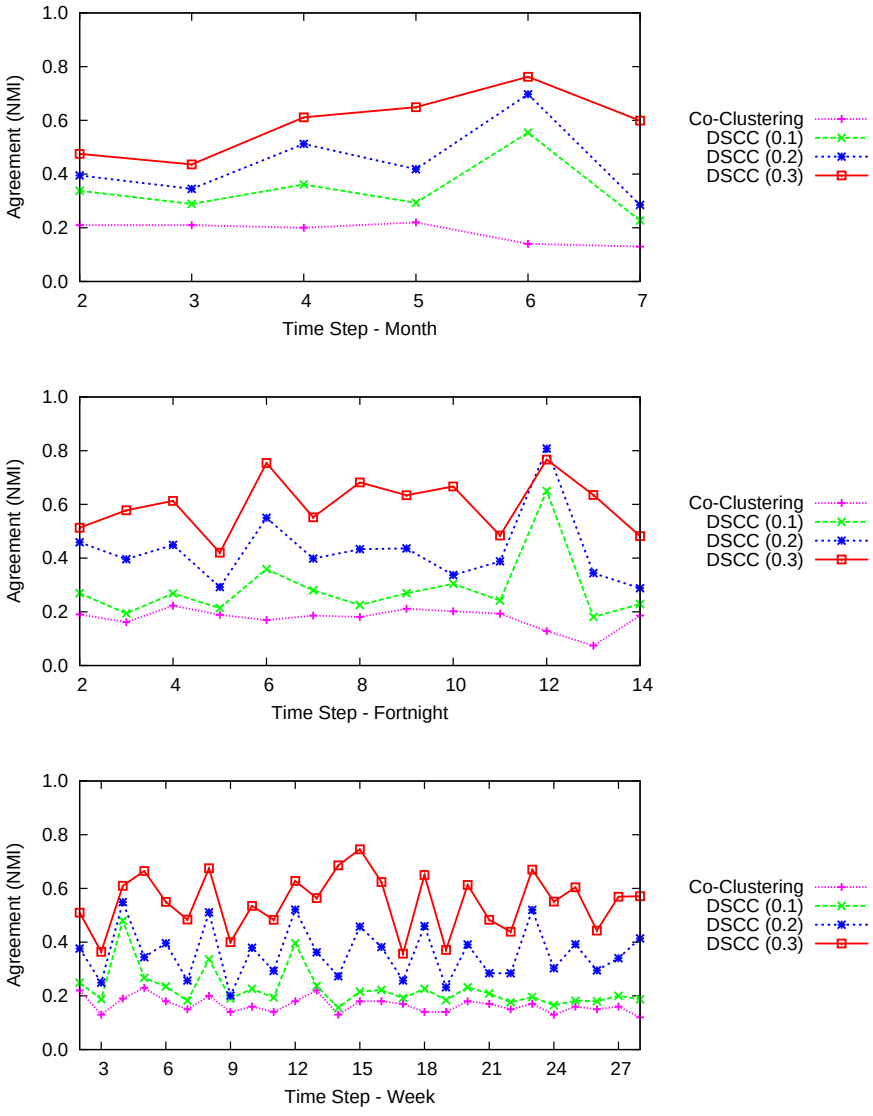


Figure 6: Comparison of agreement (in terms of NMI) between successive feature clusterings, generated by spectral co-clustering and DSCC ($\alpha \in [0.1, 0.3]$), on the *RCV1-5topic* dataset for time steps of duration one month, two weeks, and one week respectively.

4.2.2 Clustering Accuracy

To quantify algorithm accuracy, we calculated the NMI between clusterings and the relevant annotated document label information for each time step. Note that, in this case, NMI figures are only calculated based on document assignments, as annotation information is not available for terms. Figure 7 illustrates a comparison of the accuracy achieved by traditional spectral co-clustering and dynamic co-clustering on the *RCV1-5topic* dataset for the three different time step sizes.

We observed that, for monthly and fortnightly time steps, the accuracy achieved by dynamic co-clustering was not significantly higher. However, for the weekly case, there was a noticeable increase in accuracy. In the case of $\alpha = 0.3$, DSCC lead to higher accuracy on 20 of the 28 weekly graphs.

These results could appear surprising given the increases in temporal smoothness demonstrated Figure 6. However, on closer inspection, it is apparent that there is a strong *concept drift* effect in the data, as the composition of topics changes over seven months. Therefore, for longer time periods, there is a greater change in the clusters identified in successive time periods. In such cases we expect historic information to be less useful. For the shorter weekly time windows, where there is less

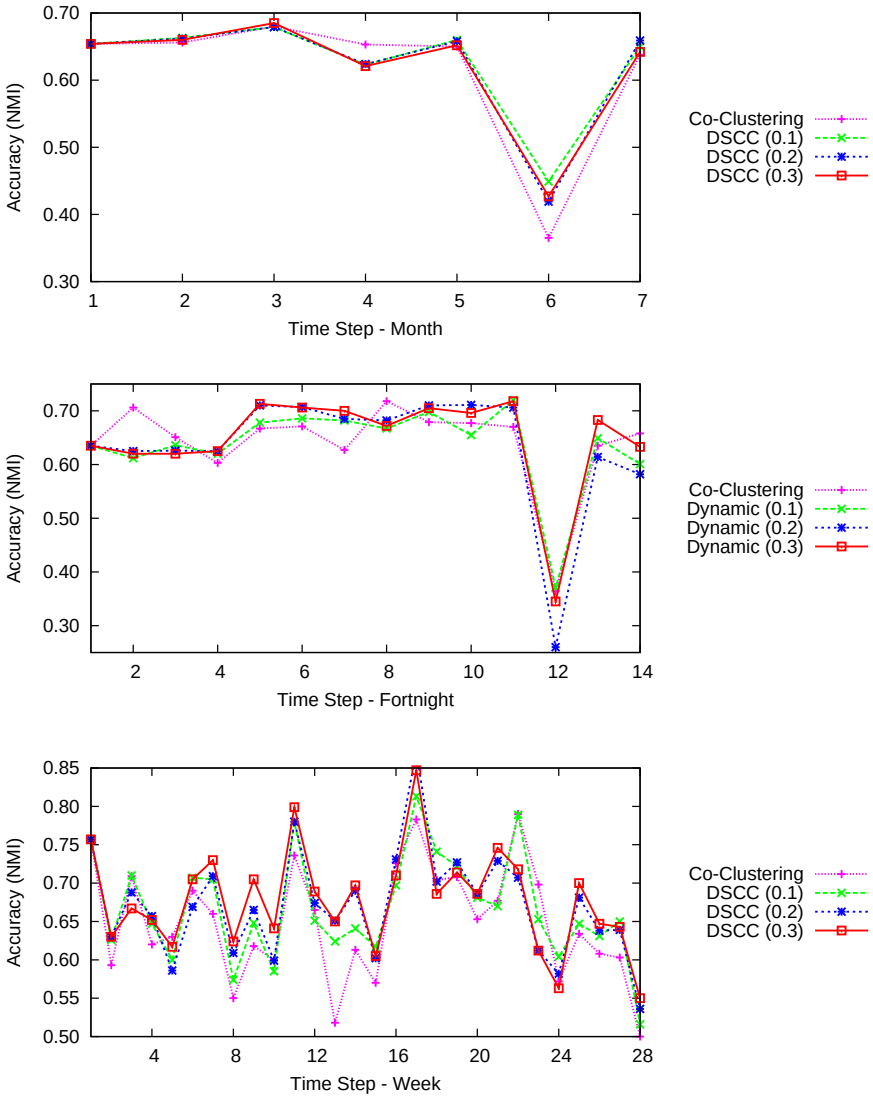


Figure 7: Comparison of accuracy for document clusterings generated by spectral co-clustering and DSCC ($\alpha \in [0.1, 0.3]$), on the *RCV1-5topic* dataset for time steps of duration one month, two weeks, and one week respectively.

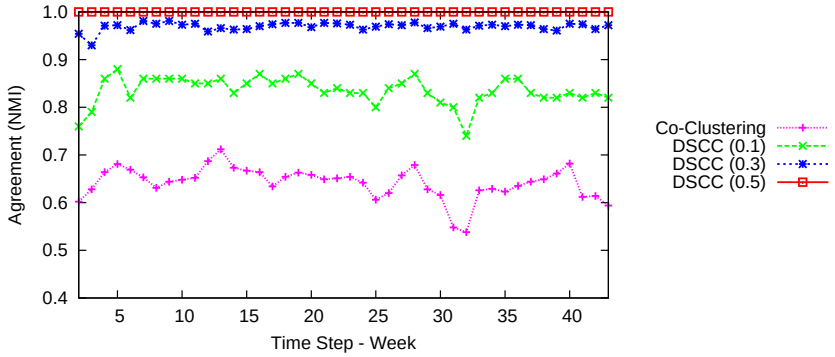
scope for drift between steps, we expect the use of historic information to improve accuracy. These results highlight the importance of selecting an appropriate time step size for step-based dynamic clustering, as has been highlighted by other researchers (Sulo *et al.*, 2010).

4.3 Evaluation on Social Bookmarking Data

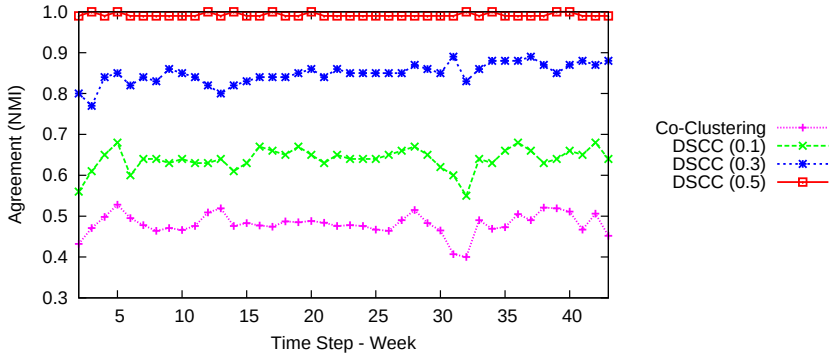
For the third phase of our evaluation, we applied the proposed co-clustering algorithm to a Web 2.0 data exploration problem. Unlike the RCV1 data, subsets of **both** objects (bookmarks) and features (tags) persist over time. We use a subset of the most recent data from a collection harvested by (Görlitz *et al.*, 2008) from the *Del.icio.us* web bookmarking portal. The subset covers the 2,000 top tags and 5,000 top bookmarks across an eleven month period from January-November 2006. We divided this period into 44 weekly time steps, and for each time step we constructed a bipartite graph – the nodes represent tags and bookmarks, and the edges between them denote the number of times each bookmark was assigned a given tag during the time step. On average, each graph contained approximately 3,750 bookmarks and 1,760 tags. For each time step, we applied dynamic co-clustering for $k_t = 20$ to identify high-level topical clusters. For these experiments, we examined a wider range of balance parameter values $\alpha \in [0.1, 0.5]$.

4.3.1 Temporal Smoothness

Figure 8 illustrates the NMI-based agreement between both tag and bookmark clusterings identified by DSCC for $\alpha = \{0.1, 0.3, 0.5\}$, when compared with the agreement scores achieved between step clusterings generated using standard spectral co-clustering (Dhillon, 2001) with $k = 20$. As with the *RCV1-topic* data, the use of historic information in DSCC leads to far more consistent clusterings between successive time steps. However, in the case of the *Del.icio.us* data this applies to both object



(a) Object (Bookmark) Clustering Agreement



(b) Feature (Tag) Clustering Agreement

Figure 8: Agreement between successive object and feature clusterings, identified by spectral co-clustering and DSCC ($\alpha \in [0.1, 0.5]$), on the *Del.icio.us* dataset across 44 weekly time steps.

and feature clusters. Again, raising the value of the balance parameter α yields increasingly smooth transitions between clusterings. In the extreme case of $\alpha = 0.5$, there is effectively no change between the predicted memberships and the final output of the co-clustering algorithm, with the iterative assignment phase converging to a solution that is almost identical to the predicted clustering.

4.3.2 Cluster Content

A number of authors (*e.g.* Berger-Wolf & Saia (2006)) have suggested analysing the stability or "loyalty" of object member-cluster memberships across time. In the bipartite case, we can quantify this for both objects and features – we suggest the latter can be used to generate meaningful labels for dynamic clusters. Dynamic clusters are constructed from timelines of step clusters produced by DSCC, as described in Section 3.3 – we use a matching threshold of $\theta = 0.2$ in all cases. For each dynamic cluster, we can then produce a ranking of features (tags) based on their respective membership stability scores. The *membership stability* for a given feature in a dynamic cluster is defined as the fraction of time steps during which the feature is assigned to a step cluster associated with that dynamic cluster. The overall feature membership stability for a dynamic cluster is defined as the mean stability for the union of all features that were assigned to that cluster in at least one time step.

Examining the range of α parameters $\in [0.1, 0.5]$, we found the trade-off afforded by $\alpha = 0.1$ lead to the most interpretable stability-based label sets. In Table 1 we show the resulting descriptive labels selected for the ten dynamic clusters that exhibited the highest overall feature membership stability, together with a suggested topic name based on the most stable tags. These descriptions highlight a range of general areas of interest covering sites frequently bookmarked by users of the *Del.icio.us* portal during 2006.

Topic	*Top 10 Tags*
Education	academic, school, mathematics, education, spanish, grammar, elearning, learning, math, translation
Web Design	usability, navigation, web, menus, html, standards, css, webstandards, tutorials, validation
Music/Video	mp3blog, youtube, television, movie, bittorrent, divx, torrent, p2p, npr, audiobooks
Shopping	clothes, t-shirt, gifts, handmade, shirts, store, clothing, fashion, crafts, shopping
Maps	world, maps, gis, geo, googleearth, geography, gps, mapping, map, googlemaps
IT	shortcuts, tweaks, wireless, opensource, support, security, troubleshooting, system, seguridad, livecd
Games/Humor	comic, worldofwarcraft, videogames, gaming, cartoon, cats, secondlife, parody, webcomics, funny
Mobile Tech	storage, mobile, pocketpc, files, file, ical, cellphone, hosting, messaging, bandwidth
Photography	fotos, stock, pictures, digital, fotografia, photography, panorama, textures, photo, flickr
Programming	developer, ajax, java, ror, regexp, rubyonrails, python, tutorial, programacion, php5

Table 1: Top 10 tags for 10 most stable clusters (in terms of feature memberships over time) identified on the *Del.icio.us* dataset by DSCC ($\alpha = 0.1$).

5 Conclusion

In this work, we have described a spectral co-clustering algorithm for simultaneously clustering both objects and features in dynamic feature-based data, represented as a sequence of bipartite graphs. The DSCC algorithm incorporates both current and historic information into the clustering process. A key aspect of the algorithm is that it is applicable in domains where objects or features alone

persist across time. In applications on dynamic synthetic, text, and real Web 2.0 data, the DSCC algorithm was successful in identifying coherent clusters, while also ensuring a smooth, consistent transition between clusterings in successive time steps.

A natural avenue of future research relates to the visualisation of dynamic co-clusterings across time, particularly in cases where informative features could be incorporated into the visualisation, such as descriptive terms or user-assigned tags.

Acknowledgements. This work is supported by Science Foundation Ireland Grant No. 08/SRC/I140 (Clique: Graph & Network Analysis Cluster)

References

Basu, S., Banerjee, A. & Mooney, R. (2002). Semi-supervised clustering by seeding. In *Proc. 19th International Conference on Machine Learning (ICML'02)*, 27–34.

Basu, S., Banerjee, A. & Mooney, R. (2004). Active semi-supervision for pairwise constrained clustering. In *Proc. SIAM Int. Conf. on Data Mining*, 333–344.

Berger-Wolf, T.Y. & Saia, J. (2006). A framework for analysis of dynamic social networks. In *Proc. 12th International Conference on Knowledge Discovery and Data Mining*, 523–528, ACM.

Bilenko, M. (2003). Learnable similarity functions and their applications to record linkage and clustering. Tech. rep., Department of Computer Science, University of Texas, Austin.

Chakrabarti, D., Kumar, R. & Tomkins, A. (2006). Evolutionary clustering. In *Proc. 12th International Conference on Knowledge Discovery and Data Mining*, 554–560, ACM.

Chi, Y., Song, X., Zhou, D., Hino, K. & Tseng, B. (2007). Evolutionary spectral clustering by incorporating temporal smoothness. In *Proc. 13th SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, 153–162.

Dhillon, I.S. (2001). Co-clustering documents and words using bipartite spectral graph partitioning. In *Proc. 7th Int. Conf. on Knowledge Discovery and Data mining*, 269–274.

Dhillon, I.S. & Modha, D.S. (2001). Concept decompositions for large sparse text data using clustering. *Machine Learning*, **42**, 143–175.

Dhillon, I.S., Mallela, S. & Modha, D.S. (2003). Information-theoretic co-clustering. In *Proc. 9th Int. Conf. on Knowledge Discovery and Data Mining*, 89–98.

Giannakidou, E., Koutsonikola, V., Vakali, A. & Kompatsiaris, I. (2010). Exploring temporal aspects in user-tag co-clustering. In *Proc. 11th International Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS)*, 1–4, IEEE.

Görlitz, O., Sizov, S. & Staab, S. (2008). Pints: Peer-to-peer infrastructure for tagging systems. In *Proc. 7th International Workshop on Peer-to-Peer Systems*.

Greene, D., Doyle, D. & Cunningham, P. (2010). Tracking the evolution of communities in dynamic social networks. In *Proc. International Conference on Advances in Social Networks Analysis and Mining (ASONAM'10)*.

Jaccard, P. (1912). The distribution of flora in the alpine zone. *New Phytologist*, **11**, 37–50.

Kalnis, P., Mamoulis, N. & Bakiras, S. (2005). On discovering moving clusters in spatio-temporal data. *Proc. SSTD 2005*, 364–381.

Koutsonikola, V., Petridou, S., Vakali, A., Hacid, H. & Benatallah, B. (2008). Correlating Time-Related Data Sources with Co-clustering. *Web Information Systems Engineering-WISE 2008*, 264–279.

Lee, D.D. & Seung, H.S. (1999). Learning the parts of objects by non-negative matrix factorization. *Nature*, **401**, 788–91.

Lewis, D.D., Yang, Y., Rose, T.G. & Li, F. (2004). RCV1: A New Benchmark Collection for Text Categorization Research. *JMLR*, **5**, 361–397.

Mirzal, A. & Furukawa, M. (2010). Eigenvectors for clustering: Unipartite, bipartite, and directed graph cases. arXiv.

Ng, A., Jordan, M. & Weiss, Y. (2001). On Spectral Clustering: Analysis and an Algorithm. *Advances in Neural Information Processing*, **14**, 849–856.

Palla, G., Barabási, A. & Vicsek, T. (2007). Quantifying social group evolution. *Nature*, **446**, 664.

Strehl, A. & Ghosh, J. (2002). Cluster ensembles - a knowledge reuse framework for combining multiple partitions. *JMLR*, **3**, 583–617.

Sulo, R., Berger-Wolf, T.Y. & Grossman, R. (2010). Meaningful selection of temporal resolution for dynamic networks. In *Proc. 8th Workshop on Mining and Learning with Graphs*, 127–136, ACM.

Tantipathananandh, C., Berger-Wolf, T. & Kempe, D. (2007). A framework for community identification in dynamic social networks. In *Proc. 13th Int. Conf. on Knowledge Discovery and Data mining*, 717–726.

Tibshirani, R., Walther, G., Botstein, D. & Brown, P. (2001). Cluster validation by prediction strength. Tech. rep., Dept. Statistics, Stanford University.

Tseng, G.C. (2007). Penalized and weighted K-means for clustering with scattered objects and prior information in high-throughput biological data. *Bioinformatics*, **23**, 2247.

Yang, Y., Pierce, T. & Carbonell, J. (1998). A study of retrospective and on-line event detection. In *Proc. 21st International ACM SIGIR Conference on Research and development in information retrieval*, 28–36.